Package: seliNDRIx (via r-universe)

June 9, 2026

Title Package for Construction of Selection Index

Version 0.1.0

Description This package is useful for construction of selection index. This package uses the mixed and random model least squares analysis to estimate the heritabilities of the traits and genetic correlation between the traits.This uses the sire model as it is considered as random effect. The genetic and phenotypic (co)variances along with the relative economic values are used to construct the selection index for any number of traits. It also estimates the accuracy of the index and the genetic gain expected for different traits.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports dplyr, psych,

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 3.5)

LazyData true

URL https://github.com/venkatesanraja/seliNDRIx

BugReports https://github.com/venkatesanraja/seliNDRIx/issues

Repository https://venkatesanraja.r-universe.dev

Date/Publication 2025-04-03 12:31:26 UTC

RemoteUrl https://github.com/venkatesanraja/selindrix

RemoteRef HEAD

RemoteSha 6524e2039578b3e5e50dbd59afe0968199094fc7

Contents

`data` *Data set for construction of selection index*

Description

This dataset contains information used for constructing a selection index.

Usage

```
data(data, package="selINDRIx")
```

Format

A data frame with 689 rows and 7 columns:

sire Sire of the cows

farm Farm from which the data were collected

soc The season of calving of a cow

poc The period of calving of a cow

tmy Total lactation milk yield in Kg

py The peak yield in Kg

afc The age at first calving

fatyield The average fat yield

`mixed_si` *Title Construction of selection index*

Description

Title Construction of selection index

Usage

```
mixed_si(data, traits, fixed, random, economic_values)
```

Arguments

data	A data frame containing the fixed, random and traits
traits	The traits for which index values are to be estimated
fixed	The fixed effects
random	The random effects
economic_values	The relative economic values

Value

Results of selection index

Examples

```
results <- mixed_si(data = data, traits = traits,
fixed = fixed, random = random, economic_values = economic_values)
```

---

random_si *Title Construction of selection index*

Description

Title Construction of selection index

Usage

```
random_si(data_input, traits, economic_values)
```

Arguments

traits	The traits for which index values are to be estimated
economic_values	The relative economic values
data	A data frame containing the fixed, random and traits
random	The random effects

Value

Results of selection index

Examples

```
results <- mixed_si(data = data, traits = traits,
random = random, economic_values = economic_values)
```

seliNDRIx *seliNDRIx: A Package for construction of selection index using mixed and random least squares analysis*

Description

This package provides functions for mixed and random least squares analysis to generate the genetic and phenotypic parameters to be used for the construction of selection index Contains two main functions: mixed() and random() that perform the least squares analysis to generate the heritability, genetic and phenotypic correlations of the traits along with their variances and covariances

Details

The package includes the following main functions:

- `mixed()`: Mixed model least squares analysis to estimate the genetic and phenotypic parameters
- `random()`: Random model least squares analysis to estimate the genetic and phenotypic parameters

Examples

```
# Example using mixed function
# Read the data
data("data", package = "seliNDRIx")
# Define your parameters
traits <- c("tmy", "py", "fatyield")
fixed <- c("farm", "soc", "poc")
random <- c("sire")
economic_values <- c(1, 0.85, 0.65)
# Run the analysis
results <- mixed_si(
data = data,
traits = traits,
fixed = fixed,
random = random,
economic_values = economic_values
)
results
# To calculate the overall selection index for each animal
SI <- c(results$SelectionIndex)  # Selection index estimates (weights) for traits
traits <- c("tmy", "py", "fatyield")  # Define the trait columns to use
overall_index <- function(data, SI, traits) {
# Ensure the number of weights matches the number of trait columns
 if (length(SI) != length(traits)) {
   stop("The number of weights must match the number of trait columns.")
 }
 # Select only the defined trait columns and calculate the index
 data %>%
```

```
   rowwise() %>%
   mutate(Index = sum(c_across(all_of(traits)) * SI)) %>%
   ungroup()
}
# Calculate the selection index
result3 <- overall_index(data, SI, traits)
# Print the result
print(result3)
# Select the top 20% of animals with the highest selection index values
top20 <- result3 %>%
 arrange(desc(Index)) %>%  # Sort by Index in descending order
 slice_head(prop = 0.2)    # Select the top 20%
# Example using random function
# Read the data
data("data", package = "seliNDRIx")
# Run the analysis
results2 <- random_si(data,
traits = c("tmy", "py", "fatyield"),
economic_values = c(1, 0.85, 0.65))
results2
# To calculate the overall selection index for each animal
SI <- c(results$SelectionIndex)  # Selection index estimates (weights) for traits
traits <- c("tmy", "py", "fatyield")  # Define the trait columns to use
overall_index <- function(data, SI, traits) {
# Ensure the number of weights matches the number of trait columns
 if (length(SI) != length(traits)) {
   stop("The number of weights must match the number of trait columns.")
 }
 # Select only the defined trait columns and calculate the index
 data %>%
   rowwise() %>%
   mutate(Index = sum(c_across(all_of(traits)) * SI)) %>%
   ungroup()
}
# Calculate the selection index
result3 <- overall_index(data, SI, traits)
# Print the result
 print(result3)
# Select the top 20% of animals with the highest selection index values
top20 <- result3 %>%
 arrange(desc(Index)) %>%  # Sort by Index in descending order
 slice_head(prop = 0.2)    # Select the top 20%
```

Index